(1)

concept analysis

F I N A L R E P O R T

MISSILE GEOMETRY PACKAGE

SBIR CONTRACT NUMBER DAAH01-88-C-0942

MARCH 31, 1989

Performed by:

Concept Analysis Corporation

14789 Keel Street

Plymouth, MI 48170

14789 keel street plymouth, mi 48170 phone 313/455-2340

Concept Analysis Corporation (CAC) markets a general-purpose CADCAM system, CADCEPT. CAC has tailored custom CADCEPT features for Redstone, using the CADCEPT Macro language, which is a powerful tool for geometric modelling using parametric design techniques. This system was used to build a geometric model of a missile.

Task 1 was an initial problem definition task. Conversations with Mr. Ed VAughn at Redstone resulted in the submission of a sketch of a typical (unclassified) missile and specifications for input format to Redstone's SPIRITS analysis program. The goal was defined to model this missile in CADCEPT, and to demonstrate that sections could be cut and fed into SPIRITS in the proper format. It was discovered that the primary reason for interest in CADCEPT was its ability to do faceted geometric modelling, and that another system of interest, BRL CAD, from the Ballistics Research Laboratories, was of interest to Redstone personnel. It was thought at the time that BRL CAD might be superior to CADCEPT for the purpose of preparing input to SPIRITS. CAC has since acquired and implemented a copy of BRL CAD, and will make a case that CADCEPT will be a nice complement for BRL CAD, especially because it is more interactive than BRL CAD, it has a drafting capability, interfaces to machining postprocessors, and a powerful macro capability. A macro is somewhat like computer software, but is really the product of an end user with an application, written in the CADCEPT command language. The medium of representation is text, and the instructions are interpreted (not compiled) by the system, inline with other user instructions. The main difference between macro writing and other steps in the operation of CADCEPT is that macros can be saved, and parameters can be varied easily at later dates. In addition, macros have control structures similar to structured programming languages. Listings of macros for modelling the sample missile and for other purposes are given in Figures 1, 7, 9, 11, 13, 15, 18, 22 and 26.

Task 2 was the preparation of demonstration macros, which illustrate the capability to handle the geometric shapes and design problems identified, and to simulate new software required. Information necessary to commence Task 2 was received on October 31, 1988. It was decided to write macros for the following components:

- o Hemispherical nose
- o Chlinder
- o Transition piece from cylinder to rectangular parallelipiped
- o Rectangular parallelipiped
- o Group of airfoil stabilizers, modelled as symmetric Joukowski profiles in cross-section.
- o Holes in the transition piece.

It has turned out that because of a bug in the UNIX version of CADCEPT which we were subsequently forced to use because of the failure of CAC's VAX computer, and consequently the VAX version, we have not been able to demonstrate the volume subtraction necessary to pierce the holes in the transition piece.

Task two has been further elaborated to output data from CADCEPT into Redstone's engineering analysis program SPIRITS, and information on this requirement was also furnished on October 31. A program has been written implemented to "thin" section data, in order to prepare it for input to SPIRITS. A conversation with Mr. Vaughn about the subject gave information that sometimes too much data is produced by ray tracing and other programs. CAC's thinner will preserve the character of sections, while descarding insignificant data. A listing for the FORTRAN source will be found in Figure 24, sample input file in figure 23, and sample output file in figure 25. Figure 22 shows that CADCEPT is capable of interfacing with SPIRITS in the sense that it can produce a file of points in the format given us by Mr. Vaughn's October 31 submission. Figure 26 shows further the reverse process, that of feeding CADCEPT with points produced by other programs. We might add that this capability is extra to the IGES capability which CADCEPT also has, in case data is produced not by just any arbitrary program, but by a CAD system which also has IGES capability.

Results of Task 2 are displayed in Figures 1-26. It can be seen from perusal of the listings that shapes can be changed considerably by the modification of a few parameters, thus illustrating parametric design, and modelling without drawings. The CADCEPT database is more than just a few interrelated files-many components are connected in logical ways. Macros allow cross checking between dimensions, so that when one dimension is changed, others can be made to change automatically.

Task 3 was the determination of software-hardware requirements. The extent of CAC's involvement in configuring a turnkey system and being in a consultative relationship was to be determined. We have determined that the target configuration is any VAX computer running ULTRIX, DEC's version of UNIX, with a Tektronix 4207-compatible terminal. Most recent information from Redstone is that the 4207's have been replaced with more advanced Tektronix terminals. We have verified that these are upward compatible with the Tektronix 4107, a slightly less powerful version of the 4207. At the time of this writing, we have

not implemented the 4107 support code for UNIX. CAC will do this implementation soon on CAC's UNIX machine, for CAC's own benefit. The use of the Tektronix terminals is straightforward and needs no special support, since they will operate in "4014 emulation" mode. The major benefit of native code support for the 4107-4207 machines as far as modeling capability of CADCEPT is concerned is that this enables a graphics tablet to be used. We understand that Redstone does not use graphics tablets. Much of the CAD world seems to be moving in the direction of iconic menus, which require only the terminal. We plan to implement icons sometime this year.

The verification of whether CADCEPT will run on the VAX under ULTRIX is being performed at the moment of this writing, and we believe that any problems will be minor, since as we understand it, DEC ULTRIX FORTRAN is more compatible with VAX-VMS than f77 FORTRAN which we are currently using. This is because VMS and ULTRIX are manufactured by the same company, Digital Equipment Corporation. The UNIX version is more portable than the VMS version, because we have eliminated the system's dependency on byte ordering. The major VAX-VMS-related system dependencies in the port to UNIX were changing the VMS "Q10" functions to UNIX system calls, which required interfacing FORTRAN routines to UNIX C-functions. These were written by the Phase I Principal Investigator prior to submission of the initial proposal for this SBIR contract.

Also as part of Task 3, we have verified by actual inspection of the code and implementation, as well as by bringing it up on CAC's in-house UNIX computer, that another CAD package of interest to Redstone, BRL CAD, is sufficiently portable and well engineered to run on Redstone's equipment. We have included the knowledge thus acquired in our Phase II proposal.

Task 4 was the preparation of system recommendations. This amounts to our Phase II report. In summary, our system recommendations are:

(1) BRL CAD, CADCEPT, and Redstone's analysis programs, such as SPIRITS, should be interfaced together in a tightly coupled fashion.

(2) Add new software to integrate the kinds of geometric entities between the packages, and to customize them further for missile geometry.

(3) Provide training and consulting services in the use of the new system.

(4) Add new three-dimensional display devices.

(5) Implement an iconic menusystem throughout.

(6) Interface to the Eagle system, a large integrated system in use at Eglin Air Force Base for Computational Fluid Dynamics. CAC has acquired a copy of this and is developing expertise in its use. CAC personnel, Dr. Glance in particular, have experience in Computational Fluid Mechanics. This, combined with a general knowledge of and commercial practice in, continuum mechanics, show that CAC is well qualified to implement the use of Eagle in connection with all the rest of the software mentioned above.

(7) Interface with CAC's structural mechanics programs, in an intelligent and specialized manner.

Task 5, presentation of results and response to review, has been done remotely, by the installation of CADCEPT on Redstone's equipment and by this Final Report and the Phase II proposal.

The following is a list of figures documenting the Phase I example problem results. "nascad" means "/nexus/nascad/nascad, my". In the following instructions, commas and periods after CADCEPT commands are for these instructions only - not for CADCEPT.

Figure 1: Listing of Macro MISSILE. This is the main macro which invokes other macros and assembles the missile. Found in nascad/repairs/missile.com . cat it.

Figure 2: Isometric view of the result of running MISSILE. Found in nascad/MISSILE.XYZ. LOAD 'MISSILE.XYZ' and SHOW 'ISO'.

Figure 3: Zoomed top view of the result of running MISSILE, showing the Joukowski profile of the fins. While data for Figure 2 is LOADed, do SHOW 'T', FIT, SHOW, ZOOM 20 20 20 -10 -10 -1, SHOW

Figure 4: Top view of the result of running MISSILE. Figure 3, FIT, SHOW

Figure 5: Right view of the result of running MISSILE. Figure 4, SHOW 'R', FIT, SHOW

Figure 6: Front view of the result of running MISSILE. Figure 5, SHOW 'F', FIT, SHOW

Figure 7: Listing of macro HEMI, used to product the nose piece. Found in nascad/hemi.com . cat it.

Figure 8: Isometric of the result of running HEMI. Found in nascad/HEMI.XYZ . LOAD HEMI.XYZ . LOAD HEMI.XYZ, SHOW 'ISO', FIT, SHOW

Figure 9: Listing of macro CYL,used to produce the cylindrical part. Found in nascad/cyl.com . cat it.

Figure 10: Isometric of the result of running CYL. Found in nascad/CYL.XYZ . LOAD 'CYL.XYZ', SHOW 'ISO', FIT, SHOW

Figure 11: Listing of macro TRANPIECE, used to produce the transition piece. Found in nascad/tranpiece.com. cat it.

Figure 12: Isometric of the result of tunning TRANPIECE. Found in nascad/TRANPIECE.XYZ. LOAD 'TRANPIECE.XYZ', SHOW 'ISO', FIT, SHOW.

Figure 13: Listing of macro BOX, used to produce the base component. Found in nascad/box.com. cat it.

Figure 14: Isometric of the result of running BOX. Found in nascad/BOX.XYZ. LOAD 'BOX.XYZ', SHOW 'ISO', FIT, SHOW.

Figure 15: Listing of macro FINS, used to produce the set of eight fins. Found in nascad/repairs/fins.com and nascad/repairs/jeffins.com. cat it.

Figure 16: Isometric of the result of running FINS & JEFFINS. Not stored as complete object. Do steps in nascad/repairs/missile.com, just the ones for the fins part, except ATTACH 'MAIN' instead of 'MISSILE'.

Figure 17: Mathematical analysis for Joukowski profile.

Figure 18: Listing of macro PROFILE, for Joukowski profile. Found in nascad/profile.com. cat it

Figure 19: Typical Joukowski profile, result of running PROFILE. Found in nascad/PROFILE.XYZ. LOAD 'PF4ROFILE.XYZ, FIT, SHOW.

Figure 20: Sections of missile, using PIERCE command. Found in nascad/repairs/MISSILE.PRC. LOAD 'MISSILE.PRC', SHOW 'ISO', FIT, SHOW.

Figure 21: Missile showing section planes. Found in /nascad/repairs/MISSILE.CUT. LOAD 'MISSILE.CUT', SHOW 'ISO', FIT, SHOW.

Figure 22: Listing of macro OUTPNTS showing one method of getting points out of CADCEPT into a file in SPIRITS format. Found in nascad/outpnts.com. cat it.

Figure 23: Listing of the file of points gotten by running macro OUTPNTS on the model produced by PROFILE. Found in nascad/PROFILE.PNTS. Cat it.

Figure 24: Listing of thinner program. Found in nascad/thin, nascad/thin.f, nascad/thin.o. A standalone program, which prompts for input and output and thin tolerance.

Figure 25: Listing of set of points produced by using thinner program on file shown in Figure 22. Found in nascad/ROFILE.PNTS. cat it.

Figure 26: Listing of macro INPNTS, showing how point files can be read into CADCEPT. Found in nascad/inpnts.com. cat it.

```
cat missile.com
Y
MACRO
E MISSILE
!  THIS FILE IS missile.com
!  MAKES A MISSILE FROM COMPONENTS STORED IN COMPONENT FILES
!  GOTTEN BY LOADs BELOW.
!  FLAG 40 'ON'      !  HOLD SCREEN ON TEXT AREA FULL.
UNSA
DVAR 'ALL'
DEFORM 'MISSILE'
ADIC 'HEMI.DAT'
HEMI 20
ATTACH 'MISSILE'
ADCELI  'HEMI' 0 0 0
MSG=  ' ADDED CELL HEMI'
MSG
ADIC 'CYL.DAT'
CYL
ATTACH 'MISSILE'
ADCELI 'CYL' 0 0 0
MSG=  ' ADDED CELL CYL'
MSG
ADIC 'TRANPIECE.DAT'
TRANPIECE 5
ATTACH 'MISSILE'
ADCELI 'TRANPIECE' 0 0 0
MSG=' ADDED CELL TRANPIECE'
MSG
ADIC 'BOX.DAT'
BOX
ATTACH 'MISSILE'
ADCELI 'BOX' 0 0 0
MSG=  ' ADDED CELL BOX'
MSG
ADIC 'FINS.DAT'
FINS
ATTACH 'MISSILE'
ADCELI 'FINS' 0 0 0
ADIC 'JEFFINS.DAT'
JEFFINS
MSG=' ADDED CELL FINS'
MSG
DVAR 'ALL'
STOR 'MISSILE.XYZ'
MSG=' ALL DONE.  '
MSG

E
0
SDIC 'MISSILE.DAT'
EXIT
[repairs] 183)
```

Figure 1. Listing of macro MISSILE. This is the main macro which calls other macros to assemble the missile.

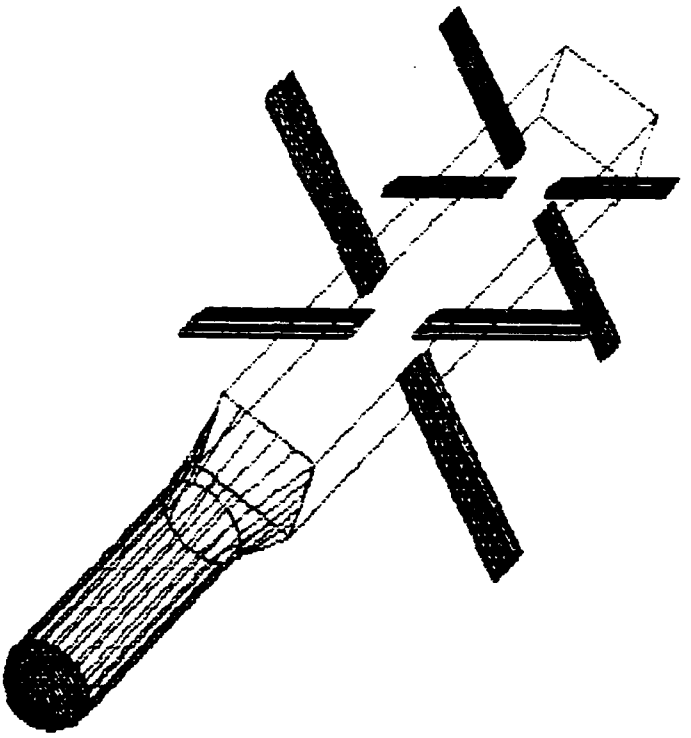

Figure 2. Isometric view of the result of running MISSILE.

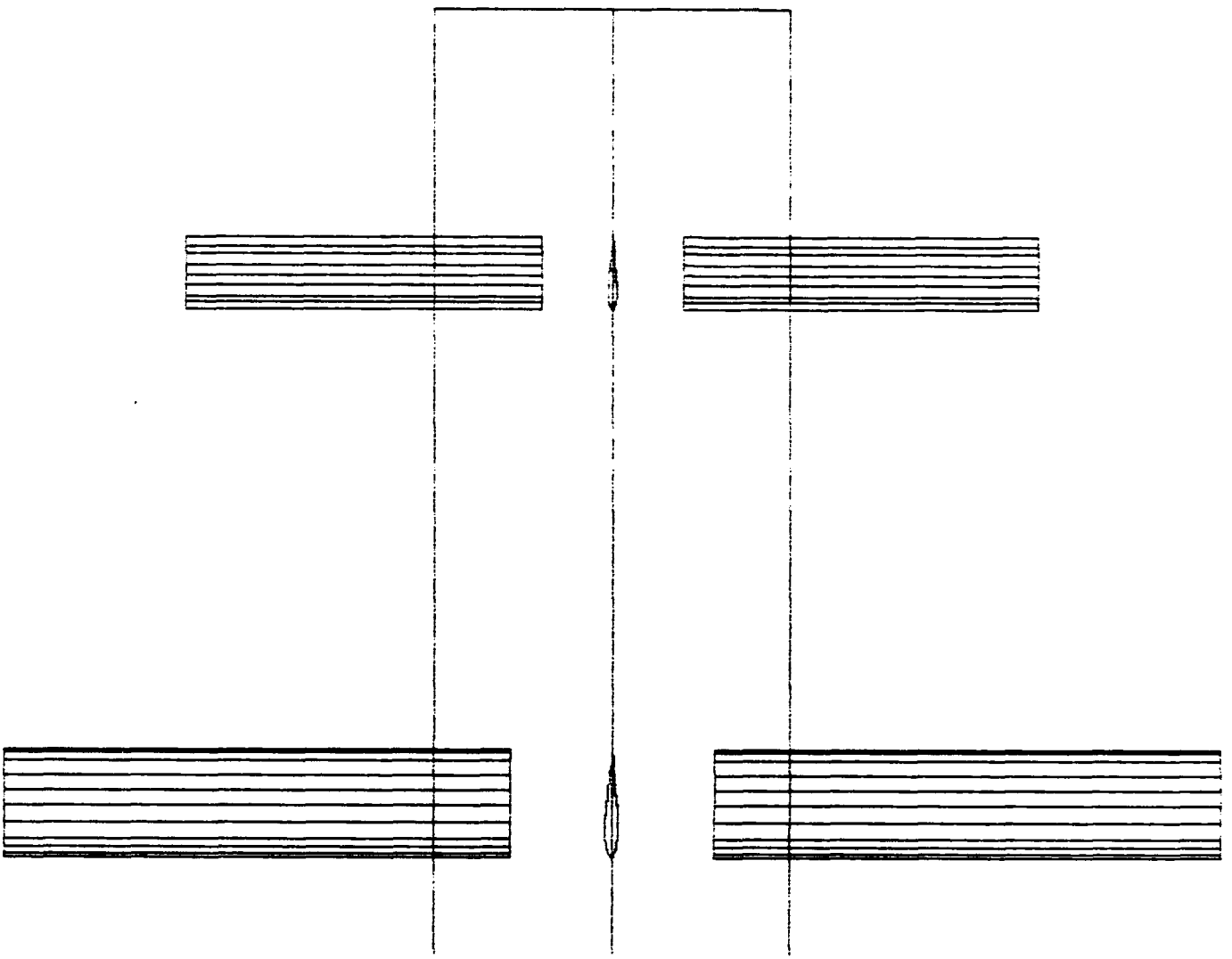

Figure 3. Zoomed top view of the rewult of running MISSILE, showing the Joukowski profile of the fins.

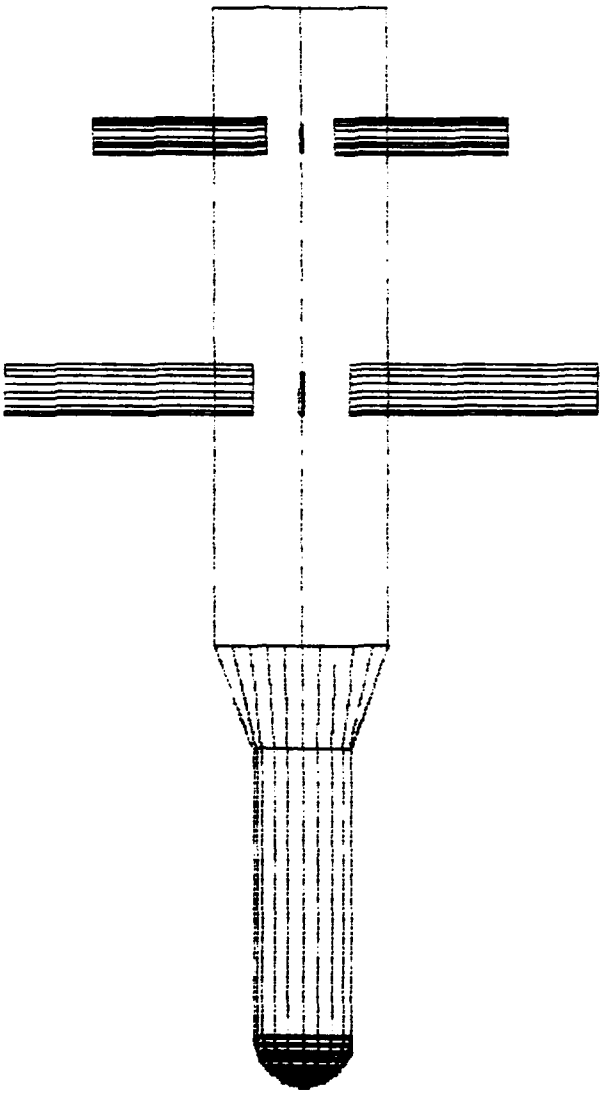

Figure 4. Top view of the result of running MISSILE.

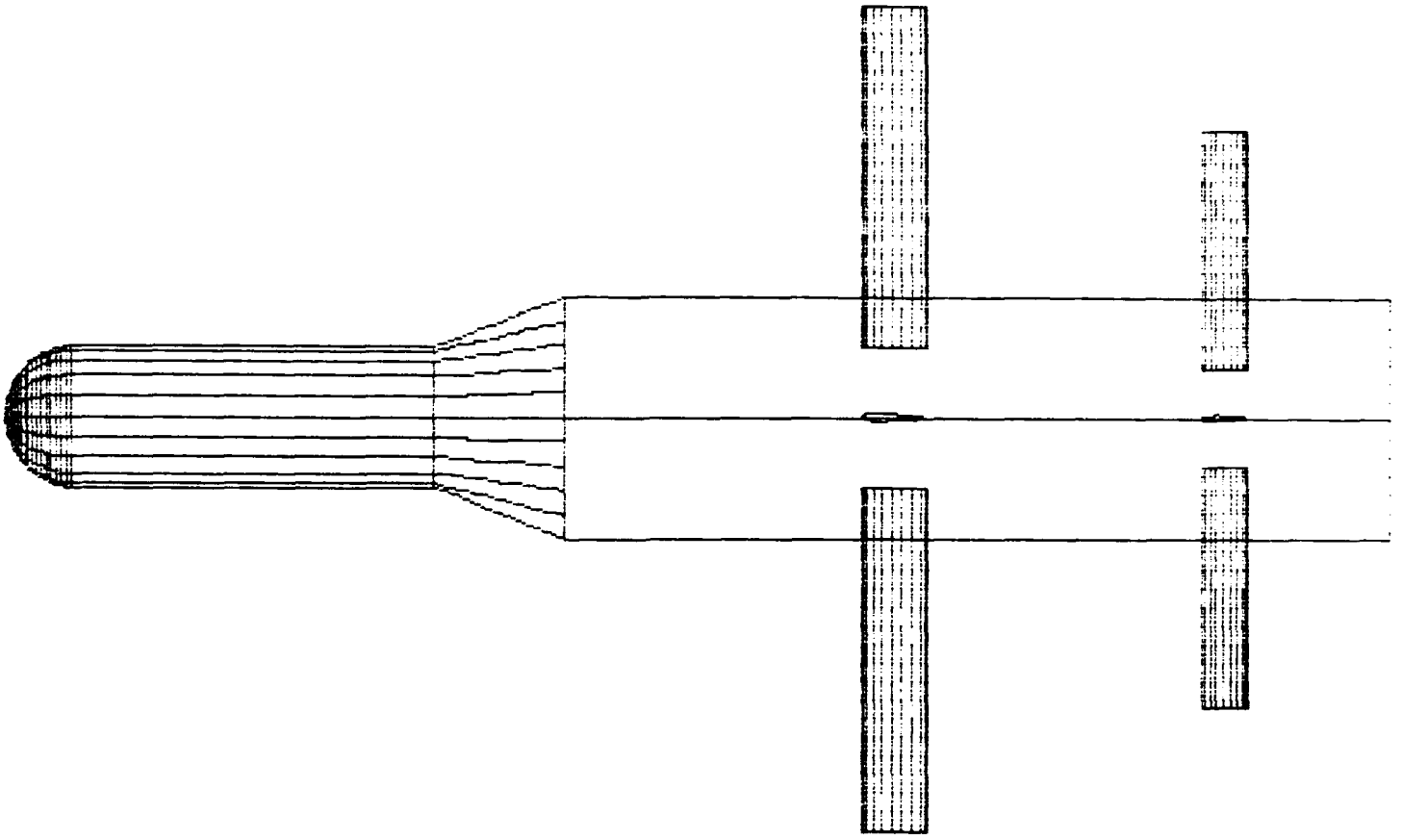

Figure 5. Right view of the result of running MISSILE.

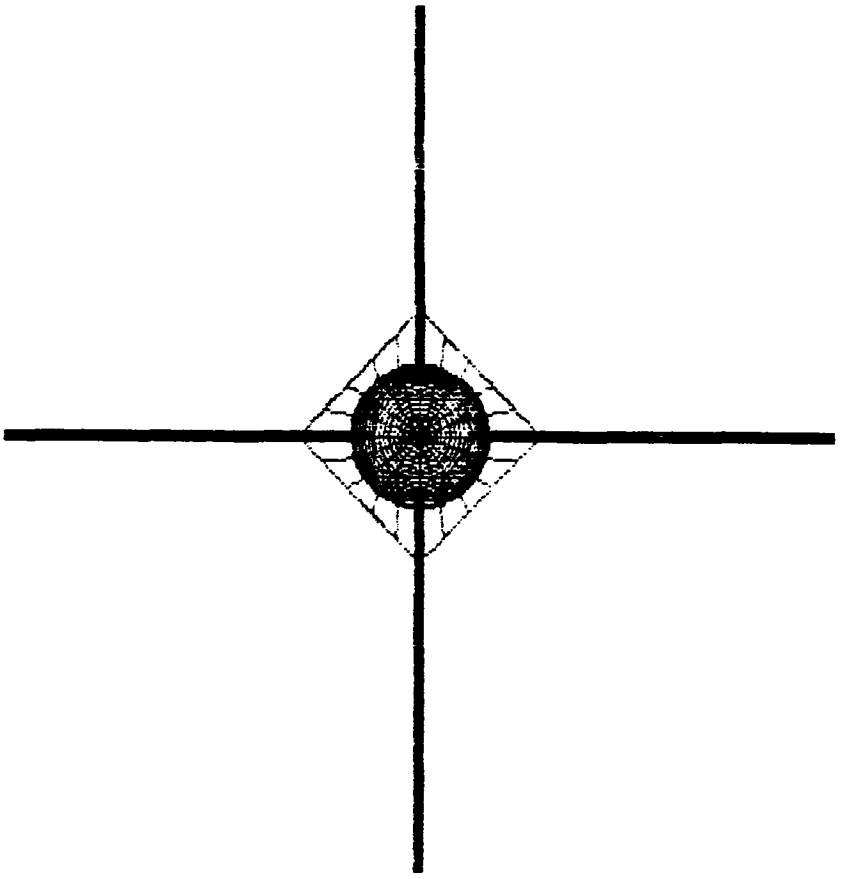

Figure 6. Front view of the result of running MISSILE.

```
cat hemi.com
/
MACRO
E HEMISTEP1
!   THIS FILE IS hemi.com
!   WILL MAKE HEMISPHERE AT CYLTOP. RADIUS CYLRAD
!   INITIALIZE PARAMETERS
FLAG 21 'ON'          !   MAKE RESULTING DRIVE SURFACE VISIBLE.
NUMSEG=#1             !   NUMBER SEGMENTS IN ALL POLYGONS.
SYS 19 NUMSEG         !   NUMBER SEGS FOR ARC DISPLAYS & POLYGONALIZATION.
SYS 2 .001
CYLTOP=0,0,80
CYLRAD=4.0
OBC

E
E HEMI
DUM=#1
HEMISTEP1 DUM           !   SET UP PARAMETERS
UNSA
DEFOPW 'HEMI'
ADDS 1
ADDA CYLTOP CYLRAD
                        !   SHOULD NOW HAVE THE "DRIVE" POLY ON LEVEL 0
UNSA
ADDS 1
!   PT1 AND PT2 WILL BE ENDS OF QUARTER CIRCLE ARC
PT11=CYLTOP(1)+CYLRAD
PT12=CYLTOP(2)
PT13=CYLTOP(3)
PT21=CYLTOP(1)
PT22=CYLTOP(2)
PT23=CYLTOP(3)+CYLRAD
PT1=PT11,PT12,PT13
PT2=PT21,PT22,PT23
ADDA PT1 CYLTOP PT2
CLEV 5
DELA
ADDS 0
DRIVE 5 0 10          !   MAKE THE HEMI.
VOLS
INWARD
OVAR 'ALL'
STOR 'HEMI.XYZ'
   MSG=' NOW EXIT, RUN INWARDPP.com, GET BACK IN, DO LODVOL & SHOW'
!   MSG

E
0
SDIC 'HEMI.DAT'
EXIT
[nascad] 155)
```

Figure 7. Listing of macro HEMI, used to produce the nose piece.

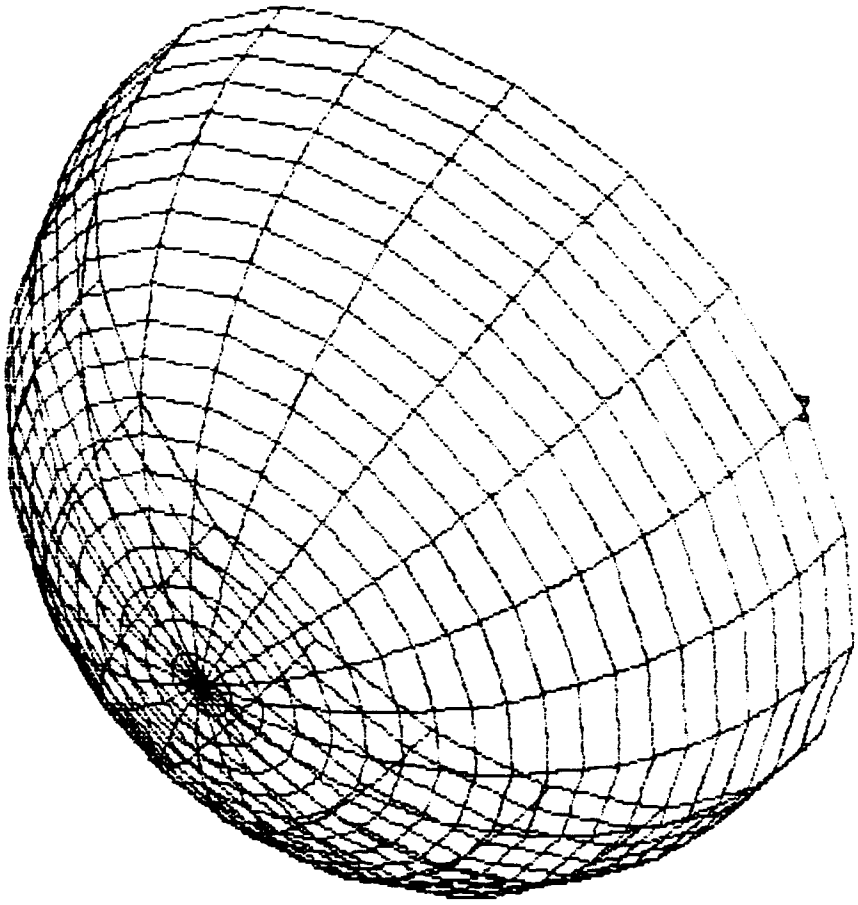

Figure 8. Isometric of the result of running HEMI.

```
[at 1/].com
:
MACRO
E CYLSTEP1
!  THIS FILE IS cyl.com
!  WILL MAKE RIGHT CIRCULAR CYLINDER.
!  INITIALIZE PARAMETERS
H1=52.          !  Z- COORD OF BOTTOM OF CYLINDER.
H2=80.          !  Z- COORD OF TOP OF CYLINDER.
CYLRAD=4.       !  RADIUS OF CYLINDER.
CYLBOT=0,0,H1
CYLTOP=0,0,H2
ALONG=CYLTOP-CYLBOT
SYS 2 ,00001
CDC

E
F CYL
CYLSTEP1
UNSA
DEFDRW  'CYL'
ADDS 1
ADDA CYLBOT CYLRAD
CLEV 0                  !  NOW HAVE LOWER CIRCLE ON LEVEL 0
COPY
MOVE ALONG
CLEV 5                  !  NOW HAVE UPPER CIRCLE ON LEVEL 5
SELA
ADDS 0
RULE 0 5 10             !  MAKE THE CYLINDER.
DELS                    !  GET RID OF THE COMPONENTS
SELA
VOLS
INWARD
DVAR 'ALL'
STOR 'CYL.XYZ'
!  MSG=' NOW EXIT, RUN INWARDPP.com, GET BACK IN, DO LODVOL, SHOW.
!  MSG

E
0
EDIC 'CYL.DAT'
EXIT
[mascad] 156)
```

Figure 9. Listing of macro CYL, used to produce the cylindrical piece.

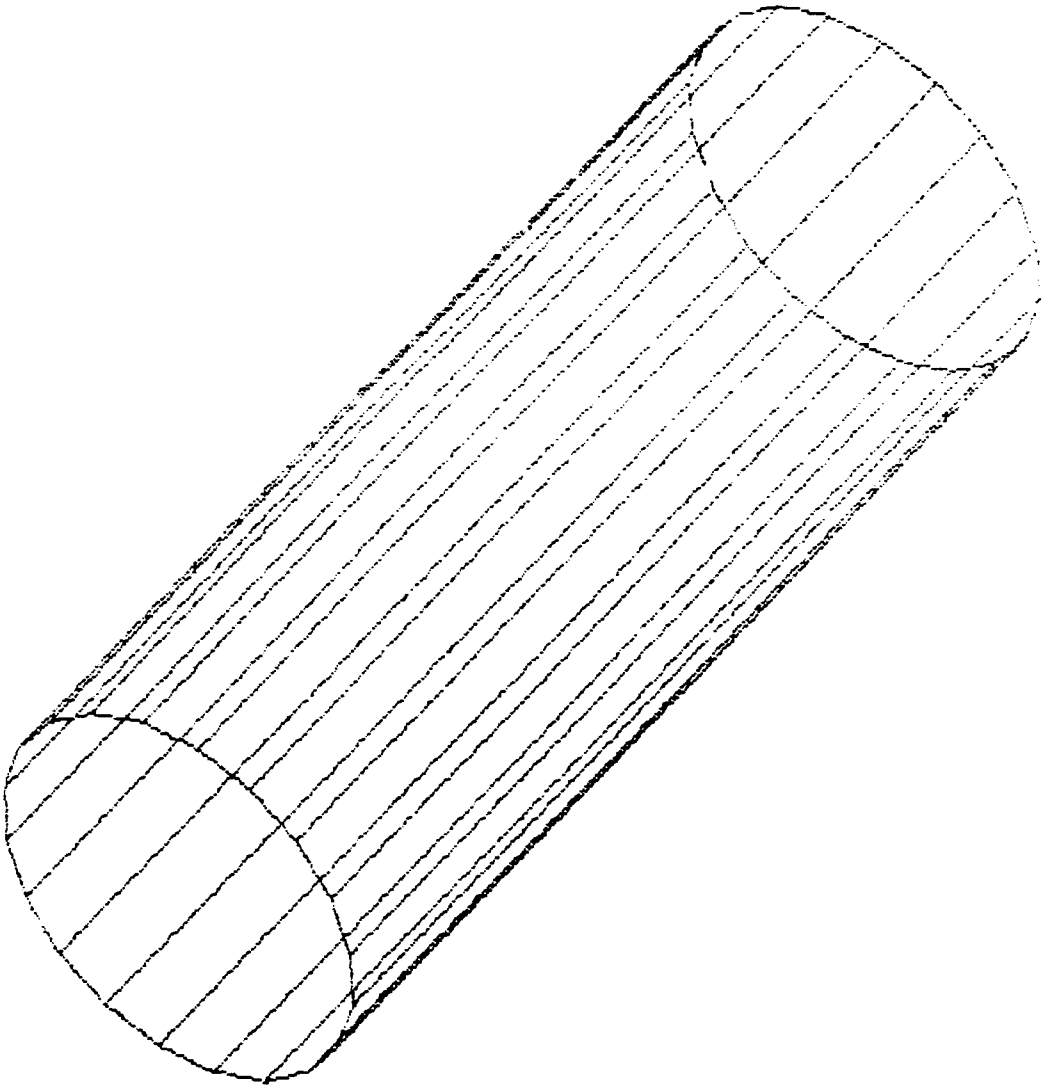

Figure 10. Isometric of the result of running macro CYL.

```
cat tranpiece.cds
'
MACRO
E TRNSTEP1
!  INITIALIZE PARAMETERS
H1=58          !  HEIGHT OF CIRCLE
H2=50.         !  HEIGHT OF SQUARE
CYLRAD=4.      !  RADIUS OF CIRCLE
BOXRAD=7.      !  HALF- DIAGONAL OF SQUARE
BOXTOP=0,0,H2
CYLBOT=0,0,H1
P1=CYLRAD,0,H1
P2=0,CYLRAD,H1
P3=-CYLRAD,0,H1
P4=0,-CYLRAD,H1
P5=BOXRAD,0,H2
P6=0,BOXRAD,H2
P7=-BOXRAD,0,H2
P8=0,-BOXRAD,H2
NUMSEG=$1          !  NUMBER SEGMENTS IN ALL POLYGONS.
SYS 118 NUMSEG     !  NUMBER SEGS FOR ARC DISPLAYS.
PALONG=P5-P6
PINC=PALONG/NUMSEG    !  INCREMENT FOR LINE SEGS IN POLY.
PBEG=P6
SYS 2 .0001
DOC
!  FORMAT (80A1,2X,I5)

E
E TRANPIECE
!  TO MAKE MISSILE TRANSITION PIECE
DUM=$1
TRNSTEP1 DUM
DEFDRW 'TRANPIECE'
CHOPUP             !  ADDS STRAIGHT LINE POLY FROM P5 TO P6,
                   !  WITH NUMSEG SEGMENTS
SELA
COPY
ROTA -90. BOXTOP CYLBOT
COPY
ROTA -90. BOXTOP CYLBOT
COPY
ROTA -90. BOXTOP CYLBOT
SELA
CONNECT P5
DELS               !  GET RID OF THE COMPONENTS
SELA               !  WE NOW HAVE A SQUARE- LOOKING POLYGON
CLEV 0             !  ON LEVEL 0
UNSA
ADDS 1
SYS 118 4*NUMSEG
ADDA P1 CYLBOT P1 'E'
CLEV 5
SELA
ADDS 0
RULE 0 5 10
DELS
SELA
VOLS
INWARD
'  MSG=' NOW EXIT, RUN INWARDPP.com, GET BACK IN, ADIC TRANPIECE'
DVAR 'ALL'
STOR 'TRANPIECE.XYZ'
'  MSG
'  MSG=' AND DO STEPS'
```

Figure 11a. Listing of macro TRANPIECE.

```
    TYPE MSG NUMSEG

E
E CHOPUP
ADDS 0
UNSA
PLITL=.01,.01,.01
PEND=PBEG+PINC
ADDL PBEG PEND
BOXL=PEND-PLITL
BOXR=PEND+PLITL
DELV BOXL BOXR
PBEG=PEND
NTIMES=NUMSEG-1
FOR (MJ1=1:NTIMES) NOTHER PBEG PINC

E
E NOTHER
!  ADDS ANOTHER SEGMENT TO THE POLYGON
XP1=$1
XP2=$2
XP3=XP1+XP2
ADDV XP3
PBEG=PBEG+PINC
DVAR XP1 XP2 XP3

E
E STEP2
LOAD 'MISSILE.XYZ'
DUM= $1
TRNSTEP1 DUM
DEFDRW 'PART1'
LODVOL
DEFDRW 'HOLE'
!  NOW MAKE A TYPICAL HOLE
SYS 118 4*NUMSEG
HCX= (CYLRAD+BOXRAD)/2.
HCENT=HCX,0,H1
HCENTUP=HCX,0,H2
HRAD=(BOXRAD-CYLRAD)/4.
UNSA
ADDS 1
ADDA HCENT HRAD
CLEV 11
UNSA
ADDA HCENTUP HRAD
CLEV 12
SELA
ADDS 0
RULE 11 12 13
DELS          !  GET RID OF THE COMPONENTS
DVAR
STOR 'MISSILE.XYZ'
SELA
VOLS
INWARD
MSG= ' NOW EXIT AND RUN INWARDPP.com. BACK IN AND RUN STEP3'
!  TYPE MSG NUMSEG

E
E STEP3
DUM= $1
TRNSTEP1 DUM
LOAD 'TRANPIECE.XYZ'
SELA
DELS
```

Figure 11b. Macro TRANPIECE.

```
ATTACH  PART1
ADDELI BOXTOP 'HOLE'
SUBVOL  'HOLE'
STOR 'TRANPIECE.X/Z'
MSG= ' NOW EXIT AND RUN VOLUMEPP.com. BACK IN AND RUN STEP4'
!  TYPE MSG NUMSEG

E
E STEP4
DUM=$1
TRNSTEP1 DUM
LOAD  TRANPIECE.XYZ'
ATTACH 'PART1'
SUBVOL
STOR 'TRANPIECE.X/Z'

E
C
SDIC 'TRANPIECE.DAT
EXIT
[hascad] 157)
```

Figure 11c. Listing of macro TRANPIECE, concl.

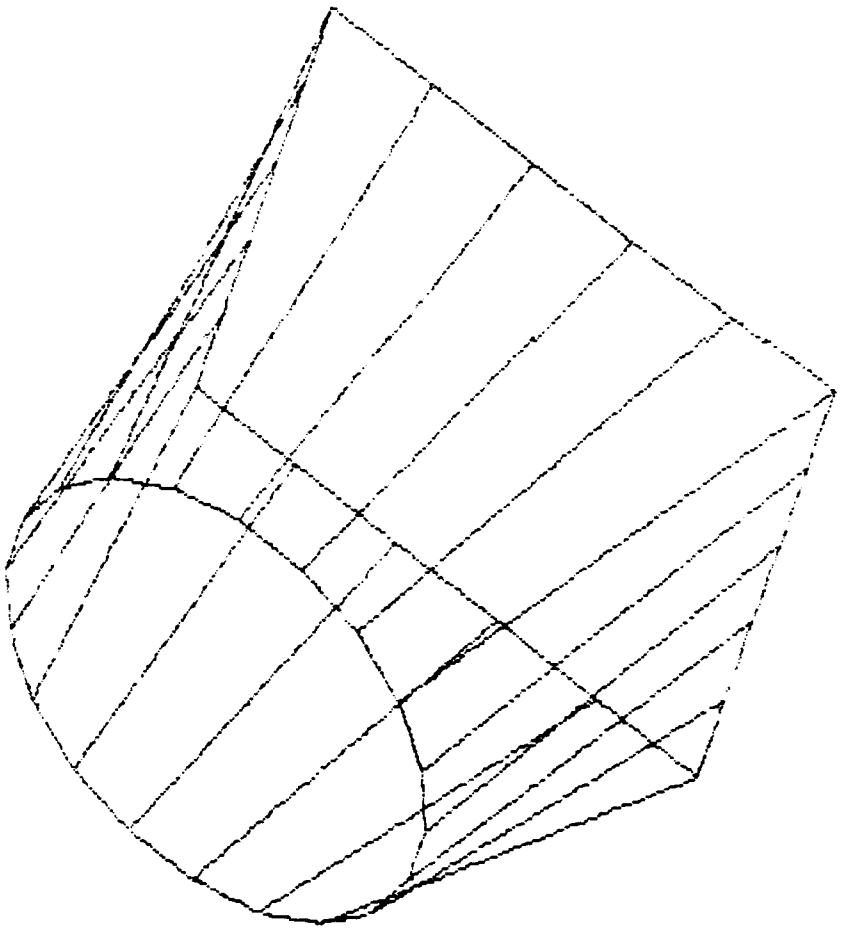

Figure 12. Isometric of the result of runnung TRANPIECE.

```
cat box.com
/
MACRO
E BOXSTEP1
'  THIS FILE IS box.com
'  WILL MAKE RECTANGULAR PARALLELPIPED.
!  INITIALIZE PARAMETERS
H1=0          !  Z- COORD OF BOTTOM OF BOX.
H2=50.        !  Z- COORD OF TOP OF BOX.
BOXRAD=7.       !  HALF- DIAGONAL OF BOX.
STLO=BOXRAD,0,H1
BOTTOM=0,0,H1
BOXTOP=0,0,H2
ALONG=BOXTOP BOTTOM
STUP=STLO+ALONG
SYS 1 .00001
DCC

/
E BOX
BOXSTEP1
UNSA
DELDRW 'BOX'
ADDS 1
ADDL BOXRAD 0 H1 0 BOXRAD H1
COPY
ROTA -90. BOTTOM BOXTOP
COPY
ROTA -90. BOTTOM BOXTOP
COPY
ROTA -90. BOTTOM BOXTOP
SELA                !  THE ROTAs DID UNSA ON OLD COMPS.
ADDS 0
CONNECT  STLO
DELS !  WE NOW HAVE LOWER SQUARE POLYGON ON LEVEL 0
SELA
COPY
MOVE ALONG
CLEV 5              !  WE NOW HAVE UPPER SQUARE POLYGON ON LEVEL 5
SELA
ADDS 0
RULE 0 5 10         !  MAKE THE PRISM.
DELS                !  GET RID OF THE COMPONENTS
SELA
VOLS
INWARD
DVAR 'ALL'
STOR 'BOX.XYZ'
!  MSG=' NOW EXIT, RUN INWARDPP.com, GET BACK IN, DO LODVOL, SHOW.'
!  MSG

E
0
SDIC 'BOX.DAT'
EXIT
[nascad] 158)
```

Figure 13. Listing of macro BOX, used to produce the base of the missile.

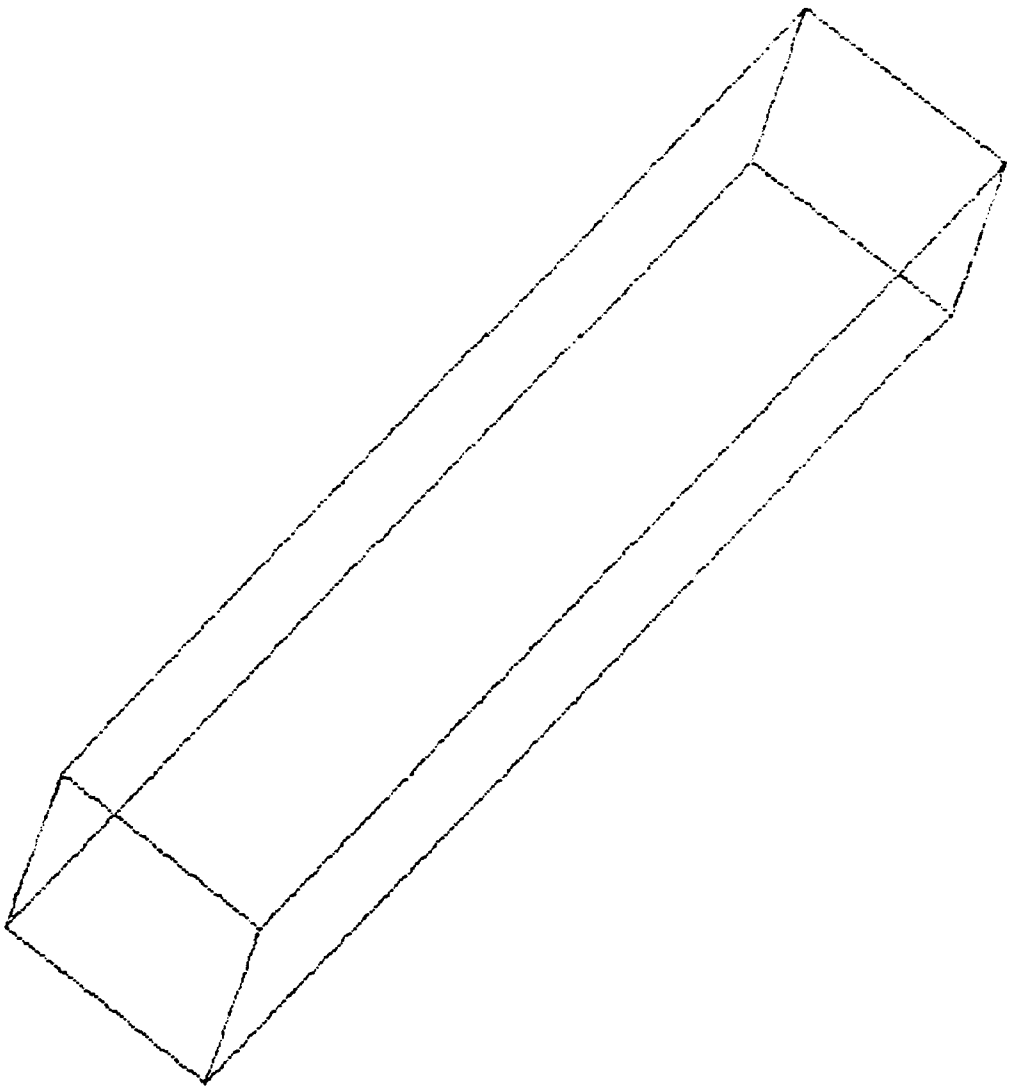

Figure 14. Isometric of the result of running BOX.

```
cat fins.com
$
MACRO
E FINSSTEP1
!  THIS FILE IS fins.com
!  WILL MAKE A MISSILE FIN
!  INITIALIZE PARAMETERS
H1=0          !  Z-PRIME COORD OF INSIDE OF BIG FINS.
H2=20.        !  Z-PRIME COORD OF OUTSIDE OF BIG FINS.
W=4           !  WIDTH
T=.6          !  THICKNESS
SMALFIN=0.0,10      !  CENTER OF SMALL FINS
BIGFIN=0,0,30       !  CENTER OF BIG FINS
MIBIGFIN=0,0,-30    !  MINUS BIGFIN
ORGIN=0,0,H1        !  -PRIME POINT OF INSIDE OF BIG FINS.
SPAN=0.0,H2         !  -PRIME POINT OF OUTSIDE OF BIG FINS
XOUT1=10,0.0        !  AXES FOR ROTATION
XOUT2=20,0,0
YOUT1=W/2.0,0,0
YOUT2=W/2.0,10,0
NUMSEG=20           !  NUMBER SEGMENTS IN ALL POLYGONS.
SYS 118 NUMSEG      !  NUMBER SEGS FOR ARC DISPLAYS & POLYGONALIZATION.
SYS 2 .001
DDC
ZVEC= SMALFIN-BIGFIN
DOWNAX=0,0,-20

E
E FINS
FINSSTEP1
UNSA
DEFDRW 'FINS'
!  TO  MAKE SYMMETRIC JOUKOWSKI PROFILE.
SQ=.19245009 !  SQRT(3)/9.0  CADCEPT DOESN'T YET HAVE SQRT.
C=W/4.0      !  CIRCLE RADIUS.
EPS=SQ*T/C   !  PROFILE THICKNESS
M=EPS*C
A=M+C
THETA= 0.0
PI=180.
TUPI=2.0*PI
THINC=TUPI/NUMSEG
FNUPT THETA
PBEG=X,Y,H1
THETA=THETA+THINC
FNUPT THETA
PEND=X,Y,H1
PLITL=.001,.001,.001
ADDL PBEG PEND
BOXL=PEND-PLITL
BOXR=PEND+PLITL
SELV BOXL BOXR
PBEG=PEND
NTIMES=NUMSEG-1
FOR (MW1=1:NTIMES) FNOTHER
SELA
ADDS 0
CONNECT
CLEV 0              !  ON LEVEL 0
COPY
MOVE SPAN
CLEV 5      !  WE NOW HAVE LOWER  POLYGON ON LEVEL 5
SELA
ADDS 0
RULE 0 5 10         !  MAKE THE PRISM
```

Figure 15a. Listing of macro FINS.

```
BE_A

E
E FNOTHER
'  ADDS ANOTHER SEGMENT TO THE POLYGON
THETA=THETA+THINC
FNUPT THETA
ADDV X,Y,0

E
E FNUPT
'H=$1
COST=COS(THETA)
DENOM=A*A+M*M-2.0*A*M*COST
C2=C*C
C2D=C2/DENOM
PMW=A*COST-M
X=PMW*(1.0+C2D)
PMW=A*SIN(THETA)
Y=PMW*(1.0-C2D)
DVAR TH DENOM PMW COST C2 C2D

E
C
SDIC 'FINS.DAT'
EXIT
[repairs] 189)
```

Fig 15b. Macro FINS, cont.

```
set _jeffins.com
/
*NOSQ
E JEFFINS
UNSA
SELI 'FINS'
MOVE 0 0 4
 ROTATE FIN INTO CORRECT WIND POSITION
ROTA 90 0 0 30 0 0 0
!ROTATE FIN TO BE PERP TO MISSILE BODY
ROTA 90 10 0 0 20 0 0
'NOW WILL COPY AND CREATE THE FOUR FINS
COPY
ROTA 90 0 0 30 0 0 0
COPY
ROTA 90 0 0 30 0 0 0
COPY
ROTA 90 0 0 30 0 0 0
UNSA
SELI 'FINS'
'NOW WILL MOVE THE FINS INTO FORWARD POSITION
!THESE FINS WILL BE THE LARGE ONES
MOVE 0 0 30
COPY
'NOW  LARGE FINS WILL BE COPIED , REDUCED IN SIZE , AND MOVED
MOVE 0 0 -20
SCAL .7 0 0 10

E
0
SDIC 'JEFFINS.DAT'
EXIT

[repairs] 190)
```

Figure 15c. Macro FINS, concl.

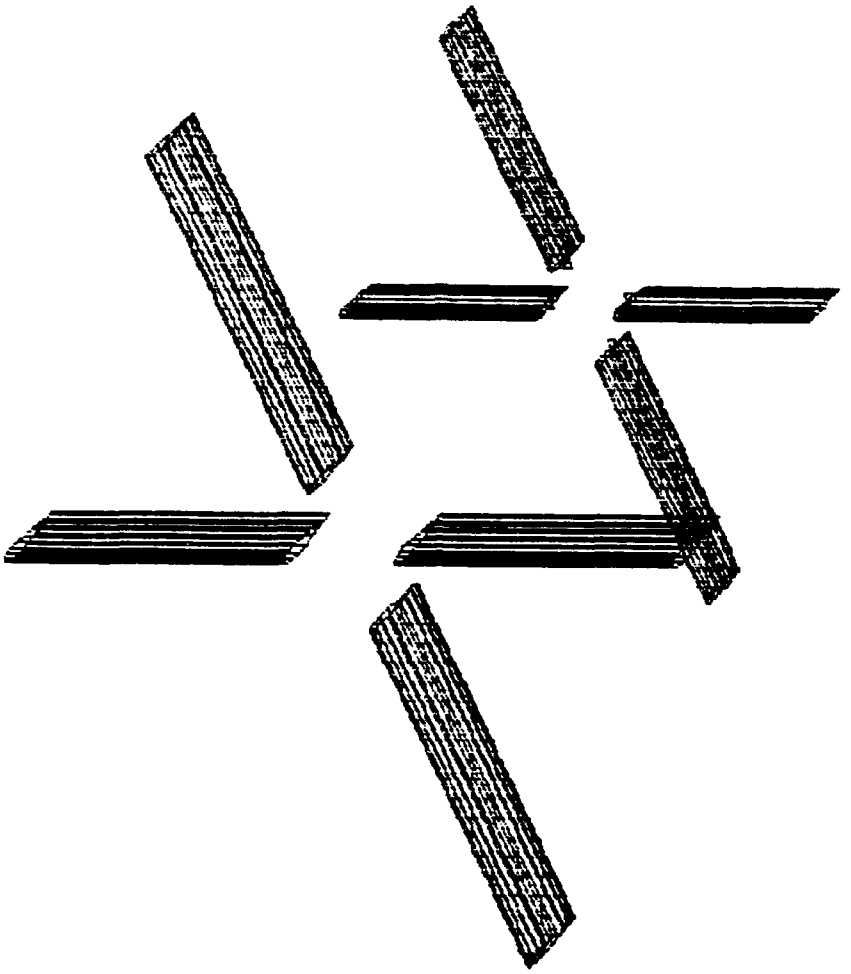

Figure 16. Result of running macro FINS.

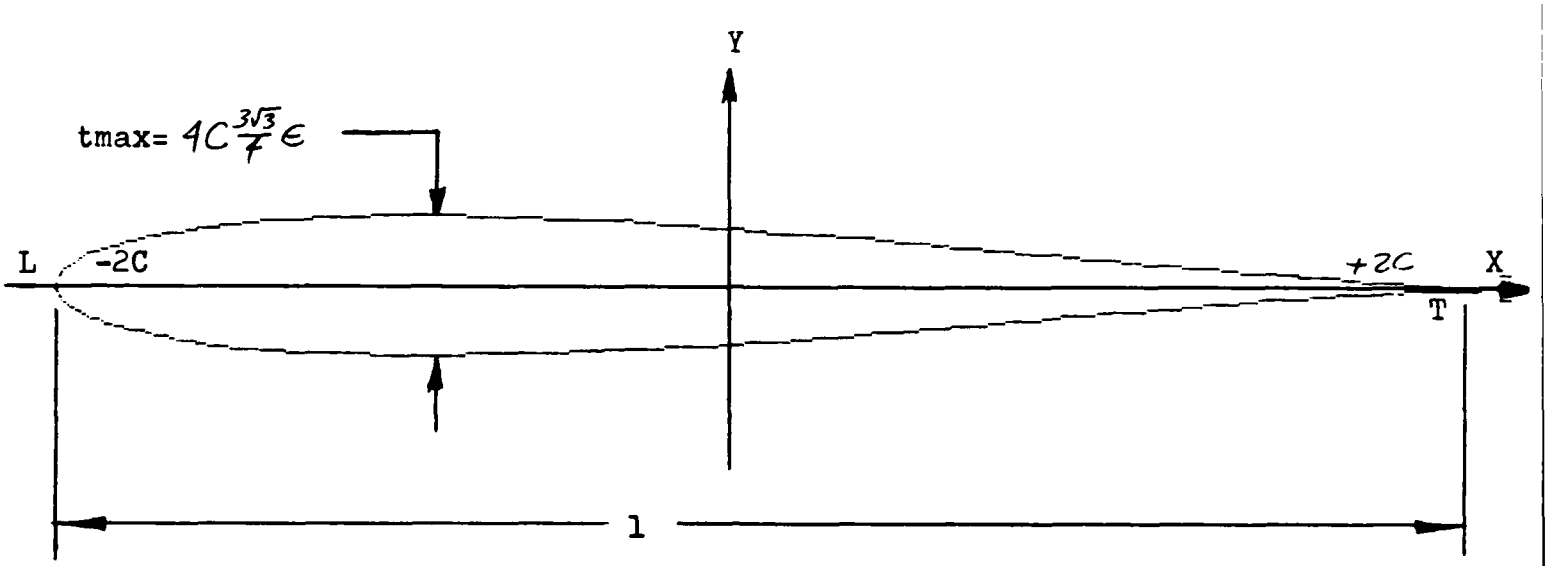

Figure 17. Mathematical analysis for Joukowski profile.

As θ goes from 0 to 2π, point (x,y) given as follows will trace out the profile:

$$x = a\cos\theta - m + \frac{c^2(a\cos\theta - m)}{a^2 + m^2 - 2am\cos\theta}$$

$$y = a\sin\theta - \frac{ca^2\sin\theta}{a^2 + m^2 - 2am\cos}$$

where, with C computed from tmax in figure, $\epsilon = \frac{t_{max}\sqrt{3}}{9C}$

$m = \epsilon C$, and $a = m + C = C(1+\epsilon)$

```
cat profile.com
'
MACRO
E PROFILE
'  TO  MAKE SYMMETRIC JOUKOWSKI PROFILE.
FORMAT (F10.3)
PROMPT 'ENTER WIDTH, F10.3' W
PROMPT 'ENTER THICKNESS, F10.3' T
NUMSEG=40      !  can't make this very much bigger-- no room.
               !  Note:  I3 format doesn't work (for prompt for this)
SQ=.19245009   !  SQRT(3)/9.0
C=W*4.0        !  CIRCLE RADIUS.
EPS=SQ*T/C     !  PROFILE THICKNESS
M=EPS*C
A=M+C
UNSA
THETA= 0.0
PI=180.
TWPI=2.0*PI
THINC=TWPI/NUMSEG
NUPT THETA
PBEG=X,Y,0
THETA=THETA+THINC
NUPT THETA
PEND=X,Y,0
PLITL=.001,.001,.001
ADDL PBEG PEND
BOXL=PEND-PLITL
BOXR=PEND+PLITL
SELV BOXL BOXR
PBEG=PEND
NTIMES=NUMSEG-1
FOR (MW1=1:NTIMES) NOTHER
STOR 'PROFILE.XYZ'
DVAR 'ALL'
MSG= ' ALL DONE'
MSG

E
E NOTHER
'  ADDS ANOTHER SEGMENT TO THE POLYGON
THETA=THETA+THINC
NUPT THETA
ADDV X,Y,0

E
E NUPT
TH=#1
COST=COS(THETA)
DENOM=A*A+M*M-2.0*A*M*COST
C2=C*C
C2D=C2/DENOM
PMW=A*COST-M
X=PMW*(1.0+C2D)
PMW=A*SIN(THETA)
Y=PMW*(1.0-C2D)
DVAR TH DENOM PMW COST C2 C2D

E
0
SDIC 'PROFILE.DAT'
EXIT
[nascad] 243)
```

Figure 18. Listing of macro PROFILE, which produces Joukowski profiles.

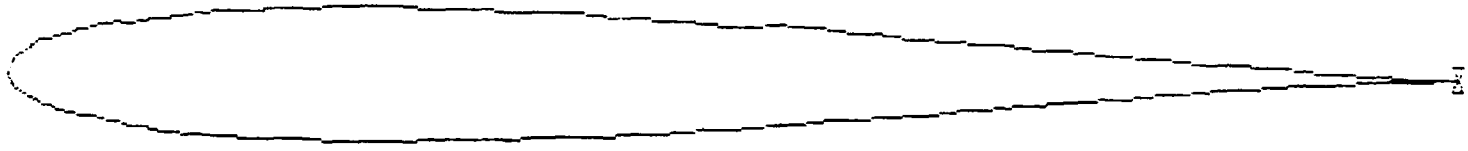

Figure 19. Typical Joukowski profile, produced by PROFILE.

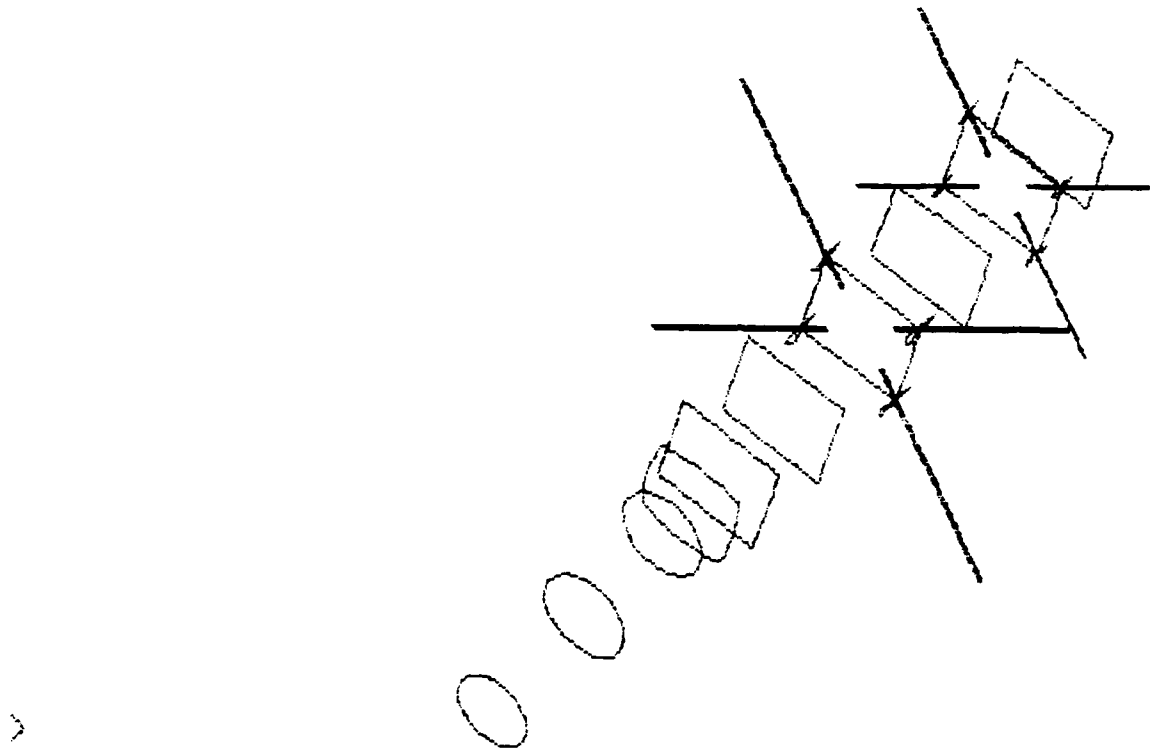

Figure 20. Sections of missile, gotten by using the CADCEPT command PIERCE. In general, we may produce input for SPIRITS this way.

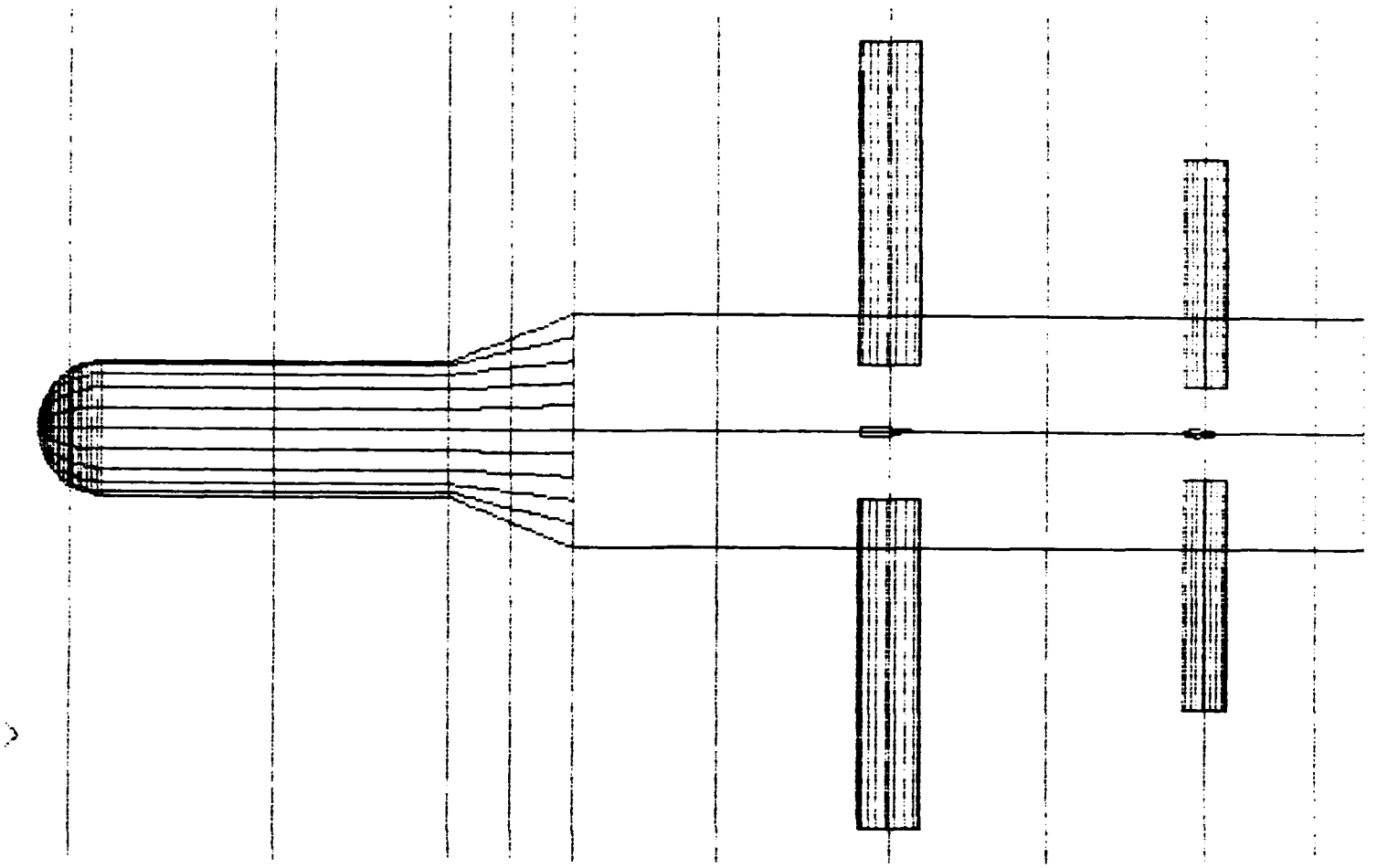

Figure 21. Showing locations of cutting planes for sections in fig 20.

```
cat outpnts.com
/
MACRO
E OUTPNTS
!   THIS FILE IS outpnts.com
!   WILL PRODUCE A FORMATTED FILE FROM A PROFILE.
OPEN 'PROFILE.PNTS'
FORMAT (3F10.3)     !   REMEMBER- FORMAT WANTS BLANK SP AFTER T.
LOAD 'PROFILE.XYZ'
FIT
SHOW
SELA
I=NXTS(0)
J=NBRE(I)
FOR (MW=1;J) DUMPT
DVAR 'ALL'
STOR 'OUTPNTS.XYZ'
!   MSG=' ALL DONE WITH OUTPNTS'
!   MSG

E
E DUMPT
PT=VRTX I.MW
WRITE PT

E
S
SDIC 'OUTPNTS.DAT'
EXIT
[nascad] 162)
```

Figure 22. Listing of macro OUTPNTS showing one method of getting points out of CADCEPT and into SPIRITS.

```
cat PROFILE.PNTS
      2.500      0.000      0.000
      2.464      0.000      0.000
      2.359      0.003      0.000
      2.187      0.011      0.000
      1.955      0.024      0.000
      1.672      0.044      0.000
      1.346      0.070      0.000
      0.987      0.100      0.000
      0.607      0.132      0.000
      0.215      0.163      0.000
     -0.179      0.191      0.000
     -0.564      0.214      0.000
     -0.933      0.228      0.000
     -1.277      0.233      0.000
     -1.590      0.227      0.000
     -1.865      0.210      0.000
     -2.097      0.183      0.000
     -2.282      0.146      0.000
     -2.417      0.102      0.000
     -2.498      0.052      0.000
     -2.526      0.000      0.000
     -2.498     -0.052      0.000
     -2.417     -0.102      0.000
     -2.282     -0.146      0.000
     -2.097     -0.183      0.000
     -1.865     -0.210      0.000
     -1.590     -0.227      0.000
     -1.277     -0.233      0.000
     -0.933     -0.228      0.000
     -0.564     -0.214      0.000
     -0.179     -0.191      0.000
      0.215     -0.163      0.000
      0.607     -0.132      0.000
      0.987     -0.100      0.000
      1.346     -0.070      0.000
      1.672     -0.044      0.000
      1.955     -0.024      0.000
      2.187     -0.011      0.000
      2.359     -0.003      0.000
      2.464      0.000      0.000
      2.500      0.000      0.000
[nascad] 193)
```

Figure 23. Listing of the file of points gotten by running the macro OUTPNTS on the file produced by PROFILE.

```
      PROGRAM USETHIN
C
C
           INTEGER FLUN, EFLUN
           REAL*8 X(3), TOL, XMAX
           REAL*8 POINTS(5,128)
           CHARACTER*32 FNAME,EFNAME
           DATA  FNAME /'                                '/
           DATA EFNAME /'                                '/
           EFLUN=12
           FLUN=11
           WRITE(6,135)
  135      FORMAT('ENTER INPUT FILENAME:')
           READ(5,136)FNAME
           WRITE(6,137)
  137      FORMAT('ENTER OUTPUT FILENAME:')
           READ(5,136)EFNAME
  136      FORMAT(A30)
           OPEN(UNIT=FLUN,FILE=FNAME(1:30),FORM='FORMATTED',
     1        STATUS='OLD',ERR=30)
           GO TO 32
   30      WRITE(6,138)
  138      FORMAT('OPEN INPUT FILE ERROR')
           GO TO 999
   32      MPTS=0
   10      READ(FLUN,101,ERR=34,END=998) X
  101      FORMAT(3F10.3)
           MPTS=MPTS+1
           DO 77 MW= 1, 3
             POINTS(MW,MPTS)=X(MW)
   77      CONTINUE
           POINTS(4,MPTS)=0
           POINTS(5,MPTS)=0
           GO TO 10
  998      CONTINUE
           TOL= 0.015
           XMAX= 100.0
           CALL THIN(POINTS,MPTS,TOL, XMAX)
           OPEN(UNIT=EFLUN,FILE=EFNAME(1:30),FORM='FORMATTED',
     1          STATUS='NEW',ERR=31)
           GO TO 33
   31      WRITE(6,139)
  139      FORMAT('OPEN OUTPUT FILE ERROR')
           GO TO 999
   33      WRITE(EFLUN,115) MPTS
  115      FORMAT(I4)
           DO 20 I=1,MPTS
   20      WRITE(EFLUN,101) (POINTS(MW,I), MW=1, 3)
  999      STOP
   34      WRITE(6, 667) MPTS
  667      FORMAT(' GOT READ ERROR. MPTS= ', I4)
           STOP
           END
```

Figure 24a. Listing of thinner program. A standalone program.

```
      SUBROUTINE THIN(X, MPTS, TOL, XMAX)
C     THINS 5D POINT ARRAY X SO THAT CHORD HEIGHT IS GREATER THAN TOL.
C     USES NO TRIGONOMETRIC FUNCTIONS OR SQUARE ROOT.
C     NPTS IS NUMBER OF POINTS IN ARRAY X.
C     THE EXTRA 2D IN X IS AUGMENTATION BY U AND V PARAMETERS.
C     WILL NOT ALLOW POINTS TO BE DISCARDED WHICH ARE CHORDALLY
C     FARTHER AWAY FROM LAST POINT THAN XMAX.  THIS ALMOST MEANS
C     THAT THE MAXIMUM CHORDAL POINT SPACING OF THE OUTPUT IS XMAX.
C     X AND XMAX ARE REPLACED ON OUTPUT BY THE THINNED CURVE, I. E.:
C     THINNING IS DONE "IN PLACE".  THIS MAY CAUSE TROUBLE ON CONVERSION
C     TO ANOTHER LANGUAGE SUCH AS C, WHERE ARGUMENTS ARE PASSED BY
C     VALUE.
C
         REAL*8 X(5,128), TOL, XMAX
         INTEGER MPTS
         REAL*8 D1(3), D2(3), CH2, CHORD, TOLSQ, Q1, Q2, Q3, DSQ
         INTEGER NPTS, L, K, LAST, L1, I, M
C     CH2 IS CONSTANT LENGTH TOLERANCE, SQUARED. NOTE THAT WE DO ALL LENGTH
C     COMPARISONS ON SQUARED LENGTHS, SINCE THE SQUARE FUNCTION IS
C     MONOTONIC. WE THUS AVOID COMPUTATION OF SQUARE ROOTS.
         CH2= XMAX * XMAX
         NPTS= MPTS
C     DONT BOTHER TO THIN LESS THAN 11 POINTS.
         IF (NPTS .LE. 10) GO TO 999
         CHORD= -1.
         TOLSQ= TOL**2
         L=1
         K=2
         LAST=1
C     PROGRAM FROM HERE ON USES FOLLOWING STRATEGY:
C     INCREMENT K UNTIL MAX CHORD HEIGHT OF RESULTING POLYGON
C     COMPOSED OF POINTS L THRU K=1 EXCEEDS TOLERANCE.
C     THEN WE SAVE THE KTH POINT AND START NEW POLYGON
C     ALSO AT KTH POINT.
    5    K= K + 1
         IF (K.GT.NPTS) GO TO 200
         DSQ= -1.
         Q1= 0.
         DO 6 M=1, 3
           D1(M)= X(M, K+1)- X(M, L)
    6    Q1= Q1 + D1(M) *D1(M)
C     DONT SAVE POINT IF VERY CLOSE TO  LAST ONE.
         IF (DABS(Q1) .LT. 1.E-8) GO TO 5
         L1= LAST +1
C     CHORD IS SQUARE OF CHORD LENGTH FROM LAST TO K.
         CHORD= (X(1, K) - X(1, LAST))**2 +
     1          (X(2, K) - X(2, LAST))** + (X(3, K) - X(3, LAST))**2
         DO 10 I= L1, K
           Q2=0.
           Q3= 0.
           DO 7 M=1, 3
             D2(M)= X(M, I) - X(M,L)
```

Figure 24b. Thinner program, cont.

```
            Q2= Q2 + D2(M) *D2(M)
     7      Q3= Q3 + D1(M) * D2(M)
C       DSQ IS DISTANCE** OF X(*, K) FROM CHORD (X(*, L), X(*, K + 1)).
    10      DSQ= DMAX1(DSQ, Q2 - Q3 *Q3 / Q1)
C       THROW OUT (DONT SAVE) POINT K
            IF ((DSQ .LT. TOLSQ) .AND. (CHORD .LT. CH2)) GO TO 5
C       SAVE POINT K
            L = L + 1
            DO 50 N= 1, 5
    50      X(N, L)= X(N, K - 1)
            LAST= K-1
            GO TO 5
   200      L= L + 1
            DO 250 N= 1, 5
   250      X(N, L)= X(N, NPTS)
            MPTS= L
   999      RETURN
            END
```

Figure 24c. Thinner program, concl.

```
cat RSFILE.PNTS
  19
      2.500       0.000       0.000
      0.987       0.100       0.000
     -0.564       0.214       0.000
     -1.277       0.233       0.000
     -1.865       0.210       0.000
     -2.097       0.183       0.000
     -2.282       0.146       0.000
     -2.417       0.102       0.000
     -2.498       0.052       0.000
     -2.526       0.000       0.000
     -2.498      -0.052       0.000
     -2.417      -0.102       0.000
     -2.282      -0.146       0.000
     -2.097      -0.183       0.000
     -1.590      -0.227       0.000
     -0.933      -0.228       0.000
      0.607      -0.132       0.000
      2.464       0.000       0.000
      2.500       0.000       0.000
[nascad] 194)
```

Figure 25. Listing of points produced by thinner program used on file of points shown in figure 22.

```
!* [.INPNTS.COM
!
MACRO
E INPNTS
!  THIS FILE IS inpnts.com
!  WILL READ A FORMATTED FILE AS A POLYGON.
!  WE JUST ASSUME HERE THAT IT HAS AT LEAST THREE POINTS.
!  THIS DEMO PROCEDURE SHOULD BE FIXED UP FOR OTHERWISE.
OPEN 'RCFILE.PNTS'
FORMAT (I4)    !  REMEMBER- FORMAT WANTS BLANK SP AFTER '.
!  FIRST READ THE FIRST TWO PTS, MAKE A LINE, SELECT THE SECOND VERTEX.
!  APPARANTLY WE HAVE TO CREATE VARIABLES FIRST BEFORE WE READ THEM.
NUMPTS= 0
READ NUMPTS
PT1=0,0,0
PT2=0,0,0
FORMAT (3F10.3)
READ PT1
READ PT2
ADDL PT1 PT2
PTINC= .001,.001,.001
PTLEFT=PT2-PTINC
PTRIGHT=PT2+PTINC
UNSA
SELV PTLEFT PTRIGHT
!  MAKE UPPER LIMIT SOME NUMBER LARGER THAN LIKELY NO OF PTs
!  IF WE MAKE THIS UPPER LIMIT VARIABLE, CADCEPT BOMBS.
FOR (MW=1:NUMPTS) MOREPT

E
E MOREPT
READ PT1
ADDV PT1   !  ADD THE POINT TO THE POLYGON WHICH WE ARE BUILDING
IF (RSTAT .EQ. 'END') MW=NUMPTS+1     !  GET OUT AT END
MSG=' IN MOREPT. MW, PT1, RSTAT= '
MSG
MW
PT1
RSTAT

E
S
SDIC 'INPNTS.DAT'
EXIT
[nascad] 195)
```

Figure 26. Listing of macro INPNTS, showing how point files can be read into CADCEPT.